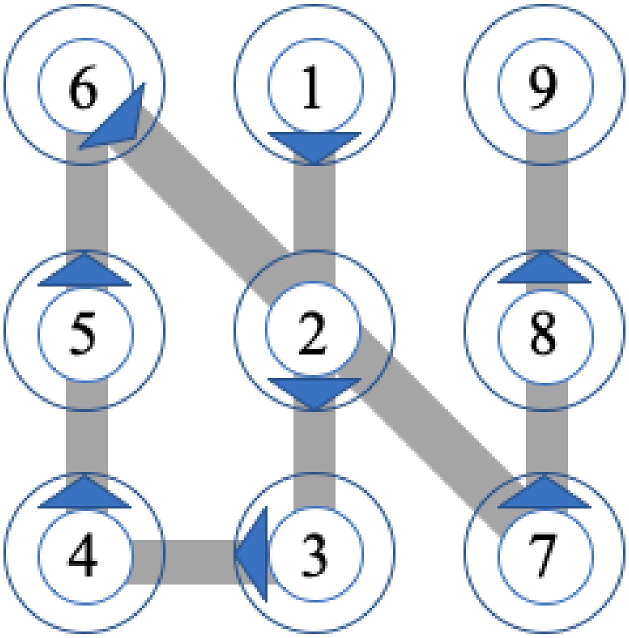
3. Ethan

Program Name: Ethan.java **Input File: ethan.dat**

Ethan has just been hired by United Intercommunication Logistics to test a new unlock pattern idea for their newest smartphone, the U-Cell 20. Their idea is to let the user choose a square grid configuration of numbers representing their password to unlock their cellular device. Research has shown that some users are more likely to remember a shape, or pattern, versus a sequence of numbers. This form of unlock pattern is known as n-pivot unlock pattern, where n is a square of an integer in the range [2,3]. Formally, an unlock pattern is a single stroke that visits each of the four or nine pivots exactly once. The pattern may start at any pivot. It may pass a pivot multiple times but only the first time counts as a visit. The phone unlocks if the pivots are visited in a predefined secret order.

Your task is to compute the distance from 1 -> 2 -> … - > n-1 -> n. The distance between two horizontal points is 1 unit and the distance between two vertical points is 1 unit.

An example, visual configuration for n= 9 is seen below, the total distance would be 9.8284 units.



Input: Input starts with a line containing an integer N (1<=N<=10), the number of test cases. Each test case begins with an integer D (2 <= D <= 3) on a single line, the dimension of the square grid to be used. The following D lines will contain D integers separated by spaces. The D x D integers represent the predefined, secret configuration that would be used to unlock the phone. All D^2 numbers will be used in the password.

Output: For each test case, output "Case # distance: X.XXXX". Where X.XXXX is the distance from:1 -> 2 -> … -> $D^2 - 1$ -> D^2 rounded to 4 decimal places. For example, for Case 1 the output would be: "`Case 1 distance: 9.8284`"

Sample Input:

```
8
3
6 1 9
5 2 8
4 3 7
2
1 2
3 4
2
1 2
4 3
3
1 2 3
6 5 4
7 8 9
3
1 8 5
7 2 4
6 9 3
3
1 6 8
2 3 4
7 9 5
3
1 2 5
6 3 4
7 8 9
2
1 4
3 2
```

Sample Output:

```
Case 1 distance: 9.8284
Case 2 distance: 3.4142
Case 3 distance: 3.0000
Case 4 distance: 8.0000
Case 5 distance: 12.0711
Case 6 distance: 13.5366
Case 7 distance: 9.2361
Case 8 distance: 3.8284
```